REMARKS

Claims 1, 2, 4-8, 10-12, 14-22, 24-28, and 31-41 were pending in the application prior to this response. Claim 1 is amended to include the limitations of dependent Claim 4 (which is cancelled) that were not shown in the cited art to more clearly define the claimed invention. Independent Claim 37 is amended to include the limitations of dependent Claim 38 (which is cancelled) to further distinguish the claimed invention from the cited references. Similarly, independent Claim 39 is amended to include the limitations of dependent Claims 40 and 41 (which are cancelled) to more clearly claim a method that is not shown by any of the art made of record.

Claims 1, 2, 5-8, 10-12, 14-22, 24-28, 31-37, and 39 remain for consideration.

Rejections Under 35 U.S.C. § 103

In the Office Action of May 21, 2003, Claims 1, 2, 4-8, 10-12, 14-22, 24-28, and 31-41 were rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,366,916 B1 ("Baer") alone or in view of U.K. Patent No. 2,347,766 ("Wilson"). This rejection is respectfully traversed based on the amendments to Claims 1, 37, and 39 and the following remarks. Initially, it should be noted that a general difference between the claimed invention and Baer is that Baer is directed to providing client applications (or a service) access to "assets" in a data store (i.e., not to a service) while the claimed invention is directed to providing access to a first service on-the-fly or dynamically to a second service (e.g., a client application that has no prior knowledge of the first service or how to interface with it).

In this regard, Claim 1 is directed to a method calling for "enabling definition of a service connector interface in conjunction with said first service" (i.e., the service being accessed) and then "subsequently invoking said service connector interface in conjunction with said second service...including instantiating said service connector interface at said second service." The method calls for an application at the second service to perform the invoking of the service connector interface and does so by "reading a configuration file providing an indicator for said service connector interface." Claim 1 requires that a service connector interface be defined for the service being accessed and that a configuration file be made available to an application that

then instantiates the service connector based on the configuration file. Further, Claim 1 calls for gaining reference to the first service by the second service by "retrieving a service instance at said service connector interface," "obtaining a service reference from said first service," and "returning said service reference obtained from said first service to said second service." Because each of these limitations is not shown or even suggested by Baer and Wilson, Claim 1 is believed allowable over the cited references.

Baer teaches using a client application layer 20 to allow a user to interact with an asset manager system (AMS) 30, which provides an interface with the "assets" in a data store 40 (see Baer at col. 3, lines 49-67). An "asset" is defined "to be a set of related data, or meta data, for a document." Baer provides no teaching of a second service gaining a reference to a first service in a dynamic manner as called for in Claim 1, but instead teaches a method of accessing data without knowing about the data layer (note, Baer does not teach that the External Services 322 and/or plug-ins 312 are unknown to the client application 20 or AMS 30). However, the AMC 200 of the client application 20 does know about the AMS 30 so it is not dynamically "gaining reference" to this "service" (see, for example, col. 4. lines 6-21 which states that the login module 2101 incorporates functions necessary to logon/logoff from the AMS 30). This is significant because the Office Action cites Baer as teaching a first service with the AMS 30 and a second service (and application program) with the client application level 20. However, it does not appear that Baer discusses a service that is being accessed by the client application but rather teaches a data store interface or connection method, which is different from the method of Claim 1.

Specifically, Claim 1 calls for "gaining reference to said first service by said second service" and then provides the specific steps of retrieving, obtaining, and returning that are not shown in Baer. The Office Action cites Baer at col. 8, lines 1-19 for teaching the gaining element of Claim 1 but at this citation, Baer appears to explain a login method 710 and how a Master Client Adapter 701 acts on a request from a Client Application to retrieve an asset from the data store 40. Baer does not suggest that a reference to a service should be retrieved, and clearly does not teach retrieving an instance at a service connector interface, obtaining a service reference from the first service, and then returning the service reference to the second service. Further, the gaining step is carried out by "an application program operative in conjunction with

said second service." This feature is not shown by Baer as suggested by the Office Action with the "Client Application" which instead teaches using the AMC 200 to manipulate a middleware application AMS 30 to access data (not a set of unknown services that are unknown by the client application and may have differing interface requirements).

The instantiating called for in Claim 1 is required to be performed by reading a configuration file providing an indicator for said service connector interface." The Office Action states that Baer is silent as to such instantiating but cites Wilson for teaching a Configuration File at page 3, lines 4-29. However, Wilson is directed toward a method of printing from the Internet using a web browser. A configuration file is used for configuring a printer, but there is no motivation in Baer to modify its teaching to use a configuration file (let alone a printer configuration file) during the instantiation of its connector adapters. Hence, Claim 1 is believed allowable because this additional limitation of Claim 1 is not shown by Baer and Baer's deficiencies are not overcome by Wilson.

Claims 2 and 5-8 depend from Claim 1 and are believed allowable for at least the reasons for allowing Claim 1. Further, Claims 6 and 8 are directed to the concept of defaulting to instantiating and then referencing a latest version or latest instance of a service being accessed or requested by client application or "second service." Wilson is cited for teaching the retrieval of a particular version. However, again, Baer does not teach obtaining a reference to and then access to a second service but only access to data in a data store. Hence, there is no discussion of versions of the data nor the need (i.e., lack of motivation found in Baer) for modifying or improving the Baer process to include the version features of Claims 6 and 8, and these claims are believed allowable for this additional reason.

Independent Claim 10 is directed to a computer program product with some limitations similar to Claim 1 but further calls for gaining reference to the first service by the second service to include retrieving an instance of the first service at the service connector interface, obtaining a service reference from the first service, and then returning said reference to the second service. Further, Claim 10 as amended requires that the first service be in "a distributed environment" and that the second service be in "a local environment" and that "the service connector interface encapsulating logic necessary to retrieve service instances in the distributed environment not in a

directory service in the local environment. Claim 10 is believed allowable for the reasons for allowing Claim 1 and also because Baer fails to teach obtaining generic services within from a distributed environment. Hence, Baer fails to teach that the first and second services are in different computing environments and that the service connector interface that is invoked within the local environment of the second service includes embedded or hidden logic necessary to obtain a reference to the first service in the distributed environment (i.e., the logic does not have to be embedded in an application in the local environment or in objects in memory at the second service). Independent Claim 10, and Claims 11, 12, and 14-18 that depend from Claim 10, are believed nonobvious in light of the teachings of Baer when combined with Wilson.

Independent Claim 19 is directed to a method similar to Claim 1 and is believed allowable for at least the reasons for allowing Claim 1 but further includes the limitation that a particular version of the first service can be specified by the second service. As discussed with reference to Claims 6 and 8, Baer fails to teach this added limitation. There is no motivation to combine Wilson with the teachings of Baer, and further, there is no teaching in Wilson that the "version" to be considered is of a service. Claim 19, and Claims 20-22 and 24-27 that depend from Claim 19, are not taught or suggested by Baer in view of Wilson and are believed in condition for allowance.

Independent Claim 28 is directed to a system for providing dynamic references between services and is believed allowable for the reasons for allowing Claim 1 (as amended to include means for instantiating a service connector at the second service) but also because it calls for means for enabling definition of a service connector interface in conjunction with the first service that includes means for developing a computer program module adhering to the service connector interface in conjunction with the first service. Claim 28 is not taught or suggested by Baer alone or in combination with Wilson and Claim 28, and Claims 30-32 and 34-36 that depend there from, are allowable over the art of record.

Independent Claim 37 is directed to a core profile engine as shown in Figure 2 with a pluggable interface for use in attaching to service modules and including a service connector associated with each attached service module. Claim 37 calls for "a pluggable interface attaching to the plug-in service modules" "wherein the pluggable interface further includes a service

connector...adapted to receive the service request from the application programming interface and to return a reference to the one service module ..." Further, as amended, Claim 37 calls for the plug-in service modules to be selected from an authorization plug-in, an authentication plug-in, a notification plug-in, a log plug-in, a group plug-in, an entity identification factory plug-in, and a replication plug-in.

The Office Action states that Baer teaches all of the limitations of Claim 37, but as discussed with reference to Claim 1, Baer fails to teach "a service connector associated with each of the attached plug-in service modules that is adapted to receive the service request from the application programming interface and to return a reference to the one service module providing the service based on the storage location." Baer fails to teach the idea of receiving a service request and responding by discovering a reference to the service and returning the reference to the requesting service. Further, Baer does not teach performing such a step based on a storage location. Additionally, the amended Claim 37 calls for specific set of service plug-ins that are not taught or suggested by Baer (or Wilson). The Office Action points to a "Plug-in Factory 708 and Plug-in Container 709 in Baer, but Baer at col. 5, line 65 states that these correspond to Plug-ins module 312 of Figure 4. The resources module 302 includes these plug-ins to allow a client adapter module 300 to "accomplish tasks required by the Client Application 20" (see col. 5, lines 19-21) but no discussion is provided on how this module operates or what the plug-ins may be. Without further explanation, Baer does not make the engine of Claim 37 obvious and does not anticipate the specific plug-ins called for in Claim 37.

Independent Claim 39 is directed to a method of providing an application with a reference to or access to a particular service. Baer in Figure 4 indicates that plug-ins 312 may be available through the AMS 30 to a client application 20. However, there is no teaching of reading a configuration file to determine an indicator to a service or of instantiating with the application a service connector for service based on the indicator. Baer instead is directed toward facilitating querying a data store 40 to retrieve data or "assets." Further, Baer does not teach operating an application to request identification of an interface implemented by the referenced service and then operating the service connector to retrieve and return the interface identification to the application for use in utilizing the reference service. Baer only mentions the "External Services" in passing as interfacing with these services is not the thrust of the Baer invention. There is no

indication that the services 322 vary over time (requiring access to new versions) or that the plug-ins 312 are not known and must be discovered (gained reference to). Because Baer was addressing a different problem (e.g., how to interface with data in a database), Baer fails to teach or suggest each and every element of the method of Claim 39. Hence, Claim 39 is believed in condition for allowance.

Conclusions

Based on the above remarks, all pending claims are believed to be allowable over the above-discussed references. Consequently, the case is believed to be in condition for allowance, and this action is respectfully requested.

No fees are believed due with this response, but any fee deficiency associated with this submittal may be charged to Deposit Account No. 50-1123.

Respectfully submitted,

Date 6/23/03

Kent A. Lembke, Reg. No. 44,866
Hogan & Hartson LLP
One Tabor Center
1200 17th Street, Suite 1500
Denver, Colorado 80202
Telephone: (720) 406-5378
Fax: (720) 406-5301